



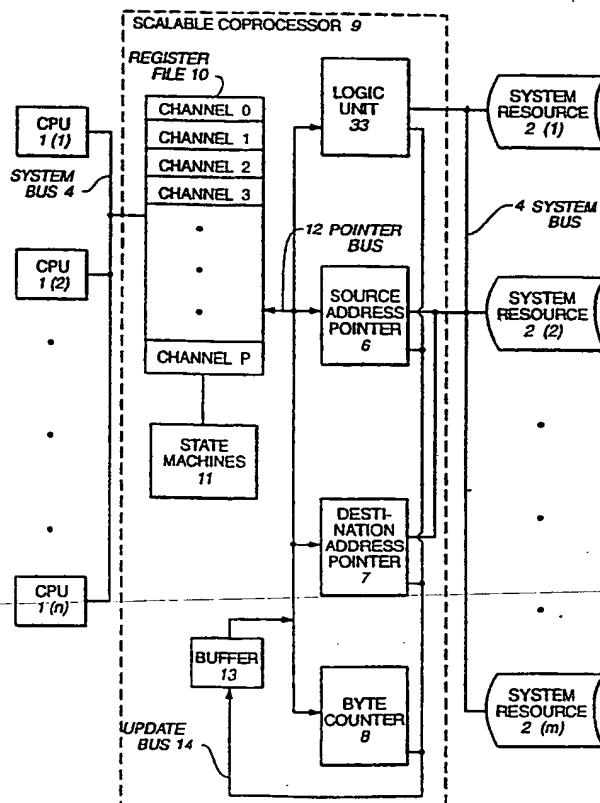
INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁵ : G06F 13/12, 13/28, 3/06		A1	(11) International Publication Number: WO 93/23810
			(43) International Publication Date: 25 November 1993 (25.11.93)
(21) International Application Number: PCT/JP93/00617		(81) Designated States: JP, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).	
(22) International Filing Date: 11 May 1993 (11.05.93)		Published With international search report.	
(30) Priority data: 07/881,299 12 May 1992 (12.05.92) US			
(71) Applicant: SEIKO EPSON CORPORATION [JP/JP]; 4-1, Nishi-Shinjuku 2-chome, Shinjuku-ku, Tokyo 163 (JP).			
(72) Inventor: KANAGALA, Sameer ; 3500 Granada Avenue, #377, Santa Clara, CA 95051 (US).			
(74) Agents: SUZUKI, Kisaburo et al.; Seiko Epson Corporation, 4-1, Nishi-Shinjuku 2-chome, Shinjuku-ku, Tokyo 163 (JP).			

(54) Title: SCALABLE COPROCESSOR

(57) Abstract

In a computing system, a scalable coprocessor (9) for enhancing communications between a set of central processing units (CPUs) (1) and a set of system resources (2). Scalable coprocessor (9) comprises a single register file (10) compartmentalized into at least two bins, each bin corresponding to a virtual coprocessor channel. Coupled to the register file (10) is a single actual coprocessor (6, 7, 8, 13, 33) for performing operations on the system resources (2). The number of virtual channels can be increased arbitrarily without the need to increase the number of actual channel hardware elements. A set of programmable state machines (11) grants operational authority to the virtual channels in the order desired and for the durations desired. Embodiments of the present invention include a fly-by DMA controller (23), an RAID coprocessor (29), and a striping coprocessor (23).



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	FR	France	MR	Mauritania
AU	Australia	GA	Gabon	MW	Malawi
BB	Barbados	GB	United Kingdom	NL	Netherlands
BE	Belgium	GN	Guinea	NO	Norway
BF	Burkina Faso	GR	Greece	NZ	New Zealand
BG	Bulgaria	HU	Hungary	PL	Poland
BJ	Benin	IE	Ireland	PT	Portugal
BR	Brazil	IT	Italy	RO	Romania
CA	Canada	JP	Japan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo	KR	Republic of Korea	SE	Sweden
CH	Switzerland	KZ	Kazakhstan	SK	Slovak Republic
CI	Côte d'Ivoire	LJ	Liechtenstein	SN	Senegal
CM	Cameroon	LK	Sri Lanka	SU	Soviet Union
CS	Czechoslovakia	LU	Luxembourg	TD	Chad
CZ	Czech Republic	MC	Monaco	TG	Togo
DE	Germany	MG	Madagascar	UA	Ukraine
DK	Denmark	ML	Mali	US	United States of America
ES	Spain	MN	Mongolia	VN	Viet Nam
FI	Finland				

1

DESCRIPTION

Title: SCALABLE COPROCESSOR

5 Technical Field

This invention pertains to the field of computing systems, and in particular to techniques for improving the communications between central processing units and system resources such as input/output controllers and memory.

10

Background Art

Figure 1 illustrates the conventional method by which central processing units (CPUs) communicate with system resources 2. The computer system comprises a set of m system resources 2, which can include memory and input/output controllers that are in turn coupled to input/output devices. The system comprises at least one CPU 1 for performing computational tasks, running stored programs, and communicating with system resources 2. Figure 1 illustrates a set of n CPUs 1, all of which are coupled to each other via a system bus 4, typically comprising a set of parallel data lines and a set of parallel address lines. System bus 4, for example, might have 32 parallel data lines and 32 parallel address lines. Using binary arithmetic, this is enough to address 4 Gigabytes of data at random. A bus master 34 is coupled to bus 4 and regulates access thereto.

A DMA controller 3 is associated with each system resource 2. All of the DMA controllers 3 are coupled to system

1 bus 4. On each DMA controller 3 is typically a bus arbiter 5,
a source address pointer 6, a destination address pointer 7,
and a byte counter 8. Bus arbiter 5 is a set of logic that is
duplicated on all of the other DMA controllers 3. Bus arbiter
5 5 stores priority information associated with that system
resource 2 and indicates to bus master 34 that one of the CPUs
1 wishes to activate the associated system resource 2. Bus
master 34 then determines which of the DMA controllers 3 will
be given authorization to become operational. Only one DMA
10 controller 3 can be operational at any one time.

The way that a CPU 1 communicates with a system
resource 2 is for CPU 1 to place, typically, three pieces of
information into the associated DMA controller 3: the source
address (the address where the data that are the subject of the
15 communication are to be found) is stored in source address
pointer 6; the address of the destination (location where the
data are to be sent) is stored in address pointer 7; and the
number of bytes desired to be moved is stored in byte counter
8. During the operational period, byte counter 8 decrements
20 once during each processing (clock) cycle until the count
stored therewithin reaches zero, at which point it is known
that all of the data have been moved. If CPU 1 wishes to
perform operations such as arithmetic, logic, or shift
operations on the data in addition to simply moving them, this
25 task can be performed by CPU 1 during the operational cycle.

As will be seen, the present invention offers a much
more efficient means for handling communications with the
system resources 2.

1 "RAID Coprocessor", a data sheet by Extended Systems
(date unknown), describes a device that, as does XOR pipeline
30 of the present invention, frees a CPU from parity
calculations in a RAID environment. However, this device can
5 process only one block of data at a time, whereas the present
invention can process multiple blocks of data simultaneously.

Disclosure of Invention

The present invention is a computer system comprising
10 at least one central processing unit (CPU) (1) capable of
performing operations on data stored within a set of system
resources (2). A scalable coprocessor (9) is coupled to the
CPUs (1) and the system resources (2) via a system bus (4).
Within the scalable coprocessor (9) and coupled to the CPUs (1)
15 is a single register file (10) that is compartmentalized into
at least two bins, each bin corresponding to a virtual
coprocessor channel. Coupled to the register file (10) is a
single actual coprocessor (6, 7, 8, 13, 33) for performing
operations on the system resources (2). Coupled to the
20 register file (10) is a means (11) for apportioning the
operational cycles of the actual coprocessor among the set of
virtual coprocessor channels.

The present invention offers the following major
advantages over the prior art:

- 25 1. The number of virtual channels can be increased
(scaled) without the need to increase the number of actual
channel hardware elements (such as items 5, 6, 7 and 8 of the
prior art), since there is but one scalable coprocessor (9).

- 1 2. Fewer routing resources means consistent and
high-speed routing across all virtual channels and higher
system clock speeds.
3. More efficient use is made of the system bus (4),
5 because the virtual channels operate in a time division
multiplex mode, with fairness built in.
4. The means (11) for arbitrating which virtual
channels gain operational access and for how long is
programmable.
- 10 5. Dynamic updating of the pointer memory array
(register file 10) by the host CPUs (1) without any unnecessary
timing restrictions.
6. A more efficient way of performing scatter/gather
operations (read and writes, respectively, when the data are
15 fragmented among many blocks), because scatter/gather is
performed in parallel rather than sequentially as has been
conventional.

Brief Description of the Drawings

20 These and other more detailed and specific objects and
features of the present invention are more fully disclosed in
the following specification, reference being had to the
accompanying drawings, in which:

 Figure 1 is a block diagram of the conventional prior
25 art technique of communicating among CPUs 1 and system
resources 2;

 Figure 2 is a block diagram of scalable coprocessor 9
of the present invention;

1 Figure 3 is a block diagram of state machines 11 of scalable coprocessor 9 of the present invention;

 Figure 4 is a sketch of amplitude versus time for gas pedal signal 15 of the present invention;

5 Figure 5 is a block diagram of a fly-by DMA controller 23 embodiment of the present invention; and

 Figure 6 is a block diagram of a RAID coprocessor 29 embodiment of the present invention.

10 Best Mode for Carrying Out The Invention

 Figure 2 is a block diagram of a general embodiment of scalable coprocessor 9 of the present invention. The environment is a computer system which comprises at least one central processing unit (CPU) 1. Each CPU 1 can be any active
15 processing element capable of performing an operation on a set of system resources 2. n CPUs 1 are illustrated in Figure 2. CPUs 1 are all coupled together via the same system bus 4. For example, system bus 4 may have 32 parallel data lines and 32 parallel address lines. This is enough to address four
20 Gigabytes of data at random.

 By using this invention, each CPU 1 can submit multiple I/O requests simultaneously. For example, a CPU 1 can be a file server that asks for hundreds of files simultaneously.

 System resources 2 are likewise coupled together via
25 the same system bus 4. Only one operation can be performed on system bus 4 at any given time. System resources 2 comprise, typically, memory and input/output controllers that are in turn

1 coupled to input/output devices such as disk drives, tape drives, CD ROMs, Bernoulli boxes, etc.

There is needed only one scalable coprocessor 9 in the computer system. This eliminates the duplication of devices and data paths that was common in the prior art, such as that
5 illustrated in Figure 1. Coprocessor 9 communicates with system resources 2 at speeds approaching memory bandwidth, e.g., 66 MBps in embodiments that have been built, rather than at slower CPU 1 bandwidths.

10 Scalable coprocessor 9 is coupled to CPUs 1 via system bus 4 and comprises register file 10, a set of state machines 11, and an actual coprocessor comprising at least one address pointer (e.g., source address pointer 6 and destination address pointer 7), byte counter 8, buffer 13, and (optionally) logic
15 unit 33. Pointers 6 and 7 are storage devices such as registers or programmable counters.

Register file 10 is a storage device such as a random access memory (RAM) that has been preselectedly compartmentalized into an arbitrary number of $p+1$ storage areas
20 or bins corresponding to the number of virtual coprocessor channels that the user of coprocessor 9 has decided to set up in advance. For example, $p+1$ may correspond to the number of blocks of data that are known to be present in the computer system. Each bin or channel stores at least three pieces of
25 information: the beginning address of the data that are to be used as the source for an operation, the beginning address of the desired destination for the data after the operation has been performed, and the size of the data that are the subject

1 of the operation. Additionally, information can be stored in
the channel indicating the type of arithmetic, logic, or shift
operation desired to be performed on the data.

5 The set of state machines 11 determines which channels
are allowed to become operational in which order and for how
long, based upon a programmable arbitration scheme stored
within state machines 11. When a given channel is allowed to
become operational, the stored source address from that channel
is placed into source address pointer 6 via pointer bus 12, the
10 destination address is placed into destination address pointer
7 via pointer bus 12, and the byte count is placed into byte
counter 8 via pointer bus 12. Additionally, the stored
arithmetic, logic, or shift instructions, if any, are placed
into optional logic unit 33 over pointer bus 12.

15 As coprocessor 9 is clocked through its normal
operational cycles by a conventional clock (not shown), e.g.,
at the rate of 66 Megahertz, the desired operations are
performed. The source data are addressed by means of source
address pointer 6 placing the source address over system bus
20 4. The destination address is similarly accessed by pointer 7,
again using system bus 4. If it is desired to perform an
arithmetic, logic, or shift operation on the data and not just
simply move them, logic unit 33 is invoked. Finally, byte
counter 8 is decremented once per cycle.

25 Typically, state machines 11 allow each channel to
perform a finite number of operations (corresponding to a given
finite number of clock cycles) per operational authorization.
128 cycles is a typical number. This may or may not be enough

1 cycles to permit the channel to complete its assigned tasks.
If it is enough, byte counter 8 decrements to 0 and state
machines 11 pass control to the next channel. If it is not a
sufficient number of cycles, the status of items 6, 7, 8, and
5 33 are passed over update bus 14 via buffer 13 and back to
register file 10 over pointer bus 12, so that the next time the
particular channel is granted operational authorization, it can
resume where it was interrupted. Buffer 13 is needed because
one of the CPUs 1 may be trying to initialize another channel
10 within register file 10 at the same time that the updated
information is being sent back to register file 10 over update
bus 14. Thus, buffer 13 prevents collisions of inbound and
outbound information.

Figure 3 illustrates in more detail the set of state
15 machines 11. The first state machine is a programmable arbiter
20, which may comprise random access memory plus associated
logic devices. Arbiter 20 stores the programmable scheme for
arbitrating which channels are given operational access and for
how long. The programming scheme may entail the use of gate
20 arrays, EEPROMs, fuses/anti-fuses, etc. The arbitration scheme
may be any one of a number of techniques such as round robin
(cycling through the channels in order and then repeating at
the zeroeth channel), priority (giving authorization only to
channels which are flagged with certain priority bytes or
25 giving flagged channels a greater number of operational cycles
than channels not flagged), etc.

A set of channel lines CH is input into programmable
arbiter 20. These lines can originate from register file 10

1 (as illustrated) or from system resources 2. The purpose of
these lines is to indicate whether the associated channels are
active (desirous of performing operations on system resources
2) or not. The output of arbiter 20 is a line conveying the
5 number of the channel which is being granted operational access
at any given time. This signal is fed to channel initialize
and update module 21. One of the outputs of module 21 is a
register file address index, which informs register file 10
which channel is being given operational authorization. The
10 length of this index is variable, depending upon the number of
channels. For example, if there are eight channels, this index
requires three bits.

The other output of module 21 is a channel ready
signal. This signal is fed to gas pedal module 22. The
15 composite output of gas pedal 22 is a square wave 15, whose
amplitude versus time is illustrated in Figure 4. When gas
pedal signal 15 is high, this indicates the presence of a
throttle (operational) period 18, i.e., one in which operations
on the data are being performed by actual coprocessor 6, 7, 8,
20 33. When gas pedal signal 15 is low, this indicates the
presence of an idle period 19 during which no virtual channel
is allowed to be operational, but rather CPUs 1 rather than
coprocessor 9 are given access to system bus 4. The durations
of the throttle and idle periods 18, 19 are variable and
25 programmable in advance.

The transition between an idle period 19 and a
throttle period 18 is denominated as a wakeup signal 16 and is
passed to arbiter 20, which induces arbiter 20 to perform a new

1 designation of authorized channel. The transition from a
throttle period 18 to an idle period 19 is denominated as a
tired signal 17, and is passed to arbiter 20 (commanding it to
designate no channel as the designated channel). Tired signal
5 17 is also passed to channel initialize and update module 21,
inducing module 21 to instruct items 33, 6, 7, and 8 to send
their status back to register file 10. This information
becomes the beginning status for the next throttle period 18
the next time that particular channel is given operational
10 authorization by state machines 11.

Figure 5 illustrates a specific embodiment of the
present invention: a fly-by DMA controller 23. This
embodiment of scalable coprocessor 9 is used in conjunction
with non-addressable devices such as input/output controllers
15 25. Since these devices are non-addressable, one of the
pointers 6, 7, from the general embodiment can be eliminated.
Thus, a single input/output address pointer 24 is used to
indicate where in memory 26 data to be read from or written to
I/O controller 25 are stored. Pointer 24 points to the
20 beginning location in memory 26 where the data are to be read
from or written to. A separate request line 28 and acknowledge
line 27 connects pointer 24 with each I/O controller 25. A
signal is sent by I/O controller 25 over request line 28 to
address pointer 24, asking DMA controller 23 to start each byte
25 transfer. Similarly, a signal is sent from pointer 24 over
acknowledge line 27 to I/O controller 25 for each byte that is
transferred, signaling that system bus 4 is available to
perform the read or write operation. Byte counter 8 decrements

1 for each transferred byte. This process continues until the
byte count for the virtual channel becomes zero, at which time
an interrupt is issued to the associated CPU 1 if required.
The data move directly from memory 26 to the I/O controller 25
5 over system bus 4, without going through controller 23. This
is characteristic of a fly-by DMA controller, as opposed to a
flow-through DMA controller.

A second embodiment of the present invention is
illustrated in Figure 6: a RAID (redundant array of
10 inexpensive disks) coprocessor 29. In this application there
are m-1 equally sized blocks of memory 26, where m-1 is at
least 2. The output is a set of m equally sized blocks of data
that are written to a set of disk controllers 2. The mth block
is a byte-by-byte parity check on the first m-1 blocks. This
15 permits fault tolerant processing: if any one block fails,
including the parity block, all of the data can be
reconstructed from the remaining blocks.

In this embodiment, register file 10 contains but a
single destination pointer, because the writing onto the m disk
20 controllers 2 is automatically partitioned equally among the m
controllers 2. Exclusive OR (XOR) pipeline 30 is a special
case of logic unit 33. Every time a source block of data is
read in from a memory 26, an exclusive OR (XOR) is performed
byte by byte, e.g., 8 bits by 8 bits even when the words are 32
25 bits long. After all of the m-1 blocks of data have been read
in, XOR pipeline 30 contains the parity block. This is written
to the destination controller 2(m) along with the other m-1
blocks of data, which are written to the first m-1 controllers

1 2. Pipeline 30 communicates with the memories 26 and disk
controllers 2 over the data lines subset 31 of system bus 4.
Similarly, address pointer 24 communicates with memories 26 and
disk controllers 2 over the address lines subset 32 of system
5 bus 4. The byte count stored in register file 10 is the same
for each source 26, because each block of input data has the
same number of bytes. State machines 11 give operational
authority to the m-1 sources 26 sequentially.

A third embodiment of the present invention is a
10 striping coprocessor. In a hardware sense, it is the same chip
as the fly-by DMA controller 23, illustrated in Figure 5. The
striping coprocessor 23 takes as inputs the m blocks of data
that have been written onto the disk controllers 2 by the RAID
coprocessor 29 and notionally (via software) stripes these
15 blocks of data onto m I/O controllers 25. Striping is an
intentional scatter, i.e., the data are fragmented into m
equally sized blocks. The number of stripes and their widths
are based upon hardware considerations.

Devices illustrated in Figures 5 and 6 have been built
20 using FPGA (field programmable gate array) technology,
specifically 4000 series architecture of Xilinx Corporation.
Other suitable construction techniques include printed circuit
boards and ASICs (application specific integrated circuits).

The above description is included to illustrate the
25 operation of the preferred embodiments and is not meant to
limit the scope of the invention. The scope of the invention
is to be limited only by the following claims. From the above
discussion, many variations will be apparent to one skilled in

1 the art that would yet be encompassed by the spirit and scope
of the invention.

What is claimed is:

5

10

15

20

25

1

C L A I M S

1. A computer system comprising:
at least one central processing unit (CPU) capable of
5 performing operations on data stored within a set of system
resources; and
a scalable coprocessor coupled to the CPUs and to the
system resources, said scalable coprocessor comprising:
coupled to the CPUs via a system bus, a single
10 register file compartmentalized into at least two bins, each
bin corresponding to a virtual coprocessor channel;
coupled to the register file and to the system
resources, a single actual coprocessor for performing
operations on the system resources; and
15 coupled to the register file, means for apportioning
the operational time of the actual coprocessor among the set of
virtual coprocessor channels.
2. The computer system of claim 1 wherein the system
20 resources comprise memory and input/output device controllers.
3. The computer system of claim 1 wherein the number
of bins is variable and is preselected.
- 25 4. The computer system of claim 1 wherein the actual
coprocessor comprises means for performing any combination of
any arithmetic, logic, and shift operations on multiple blocks
of data within the system resources.

1 5. The computer system of claim 1 wherein the
apportioning means comprises means for determining the order
and duration for which the virtual coprocessor channels are
given operational authorization.

5

6. The computer system of claim 5 wherein the
~~determining means comprises a programmable throttle~~ which sets
the number of operations each virtual channel is allowed to
perform during each operational authorization.

10

7. The computer system of claim 6 wherein idle
periods are interspersed between throttle (operational)
periods; and

15 the system bus is free to be used by the CPUs during
the idle periods.

8. The computer system of claim 1 wherein the actual
coprocessor is fly-by DMA controller.

20 9. The computer system of claim 1 wherein the actual
coprocessor is a RAID (redundant array of inexpensive disks)
coprocessor.

10. The computer system of claim 1 wherein the actual
25 coprocessor is a striping coprocessor.

11. The computer system of claim 1 wherein the actual
coprocessor comprises a byte counter coupled to the register

1 file via a pointer bus, at least one address pointer coupled to
the register file via the pointer bus, and a buffer coupled to
the address pointer(s), the byte counter, and the pointer bus.

5

10

15

20

25

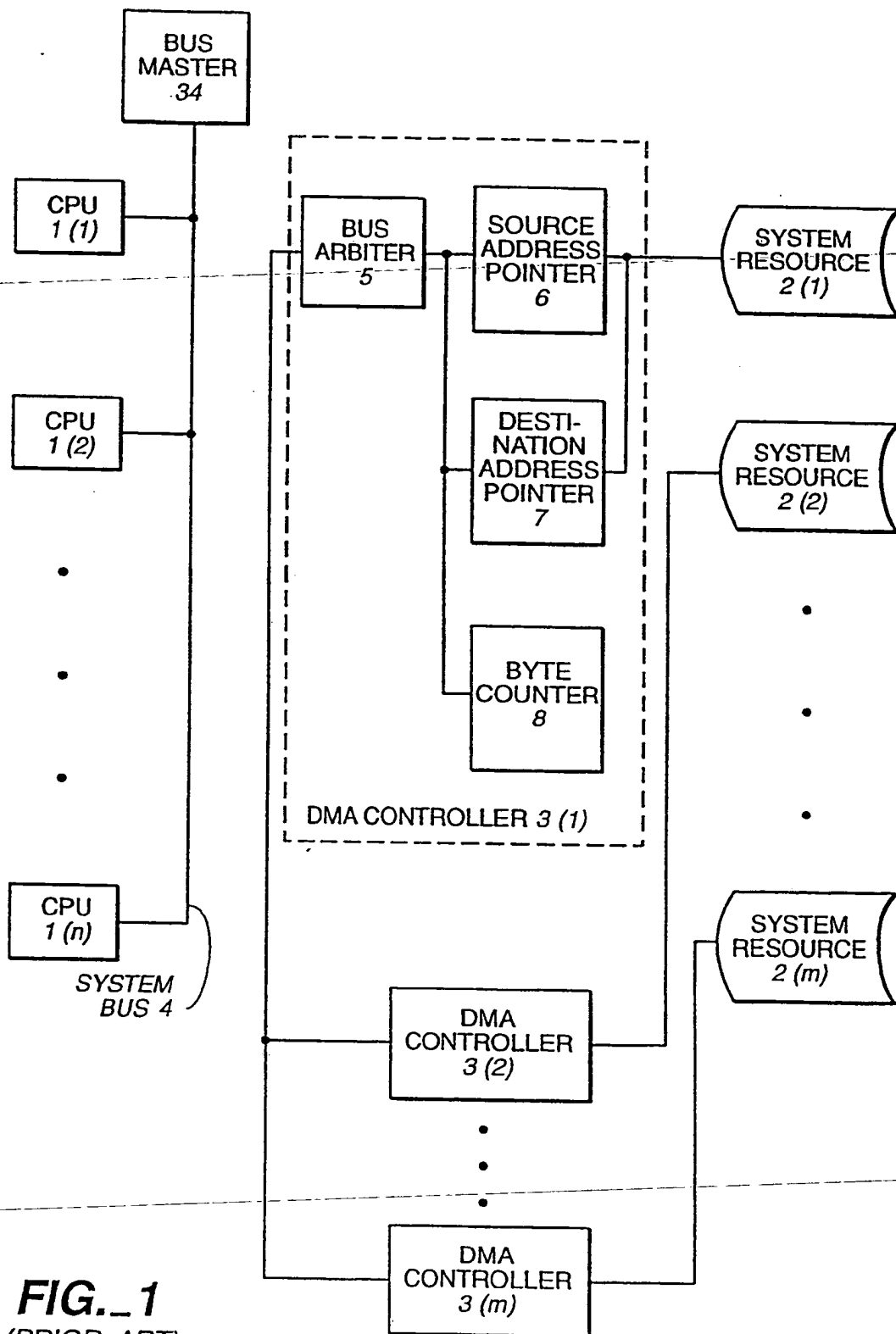
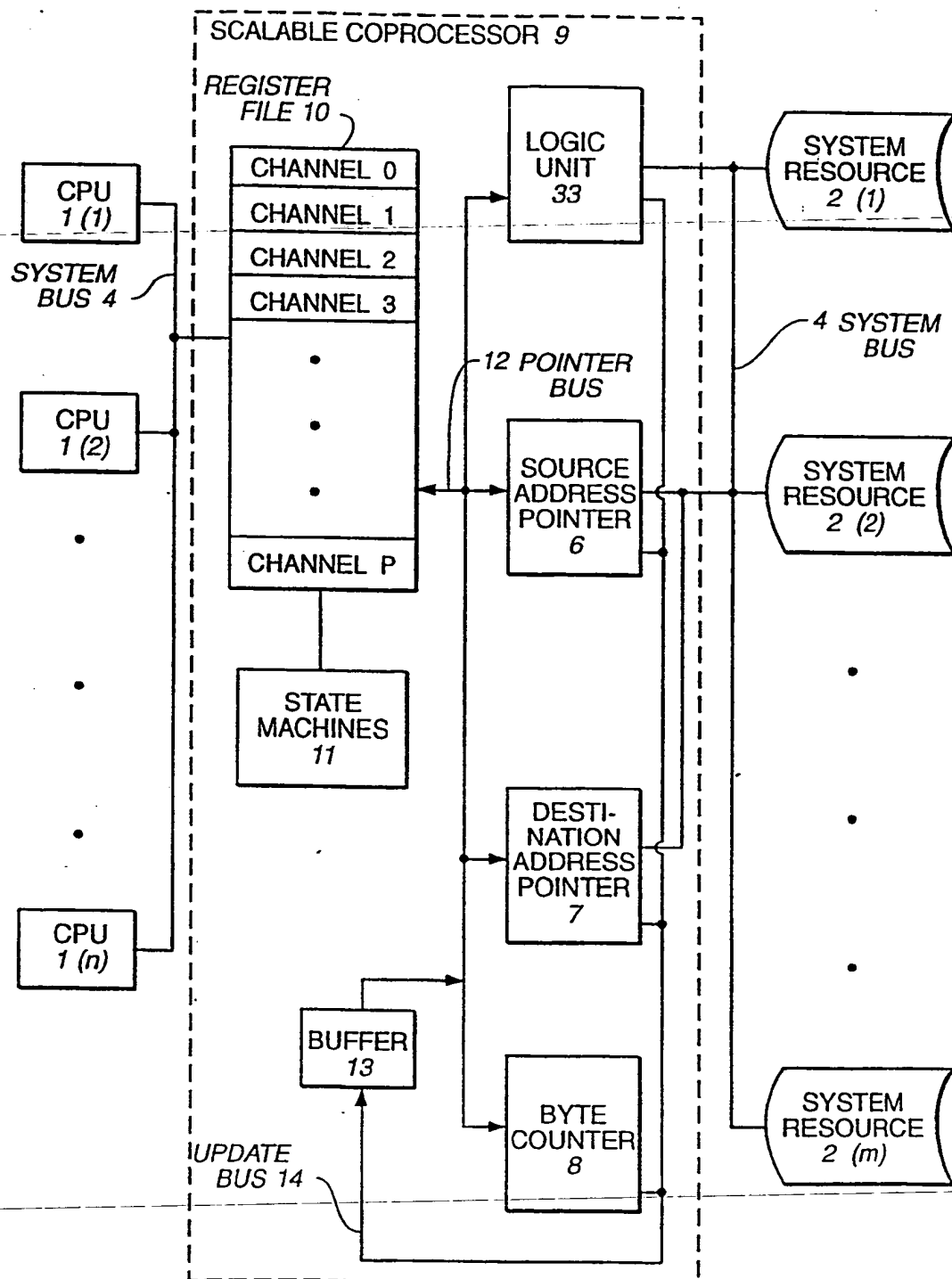


FIG. 1
(PRIOR ART)

**FIG. 2**

2 / 5

SUBSTITUTE SHEET

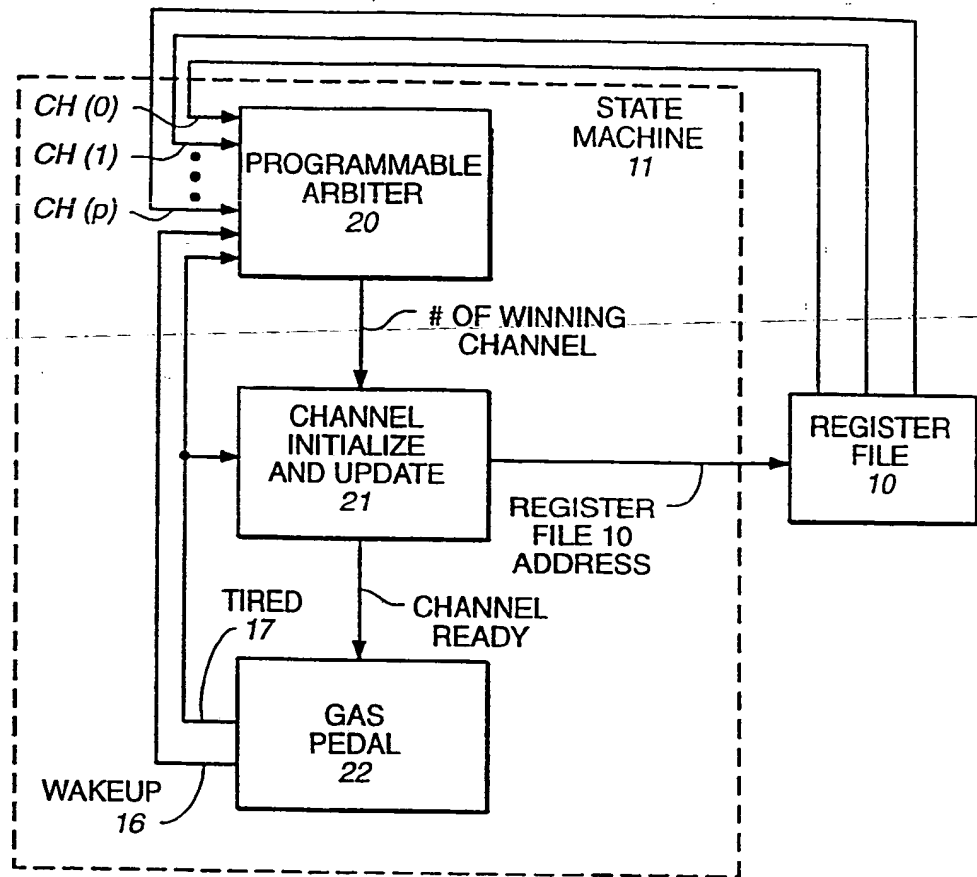


FIG. 3

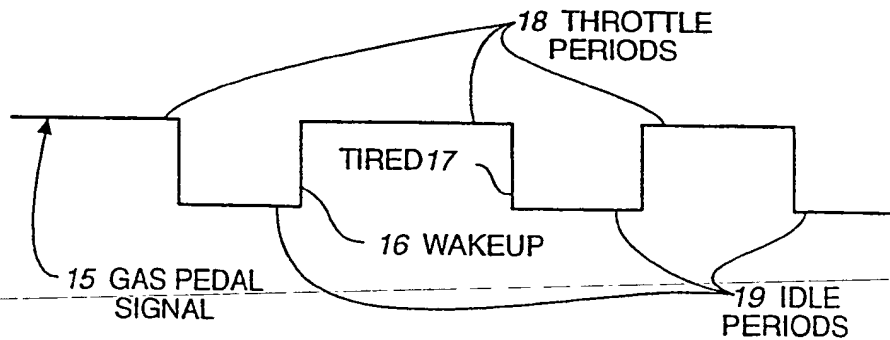


FIG. 4

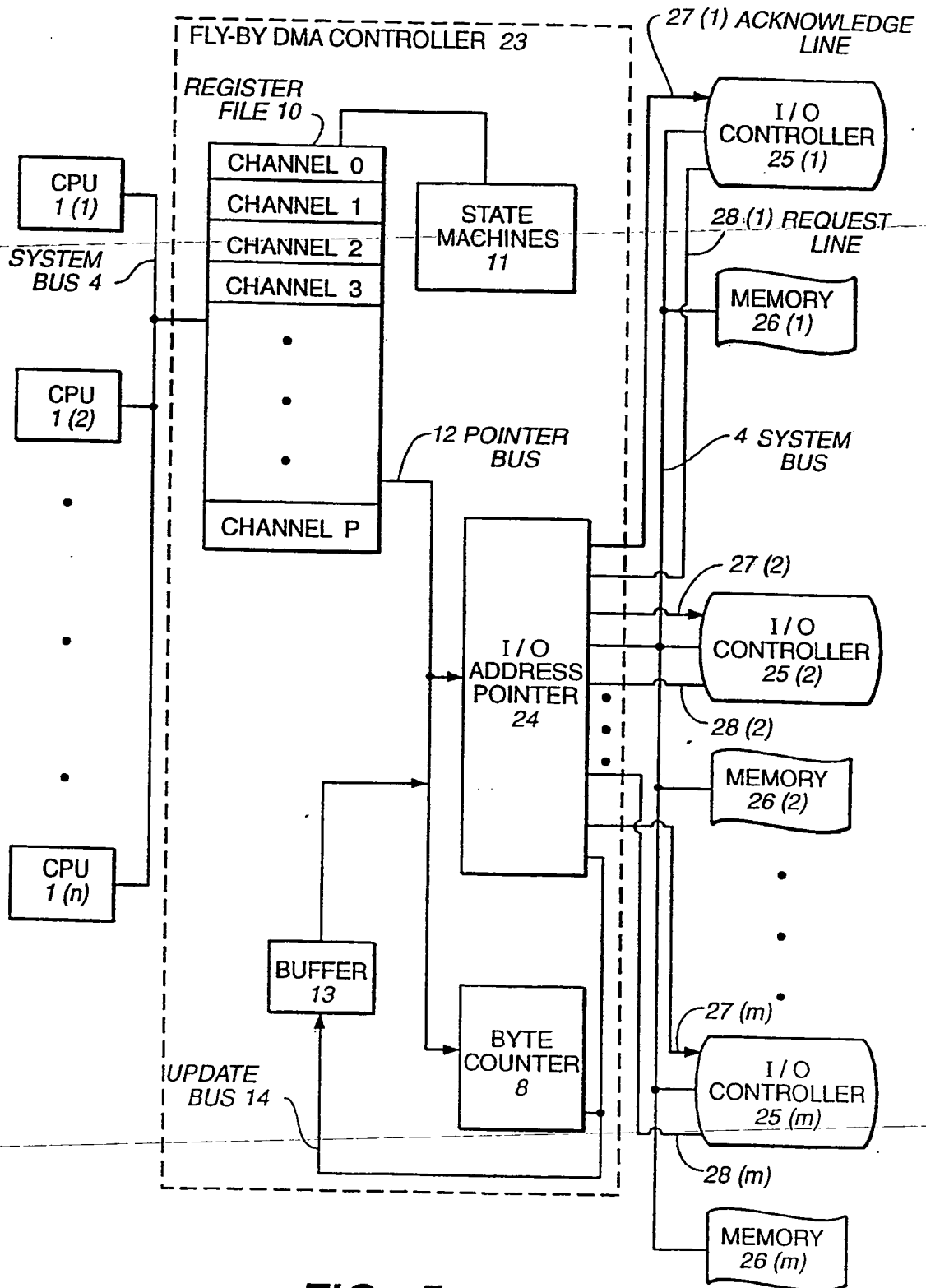


FIG. 5

4 / 5

SUBSTITUTE SHEET

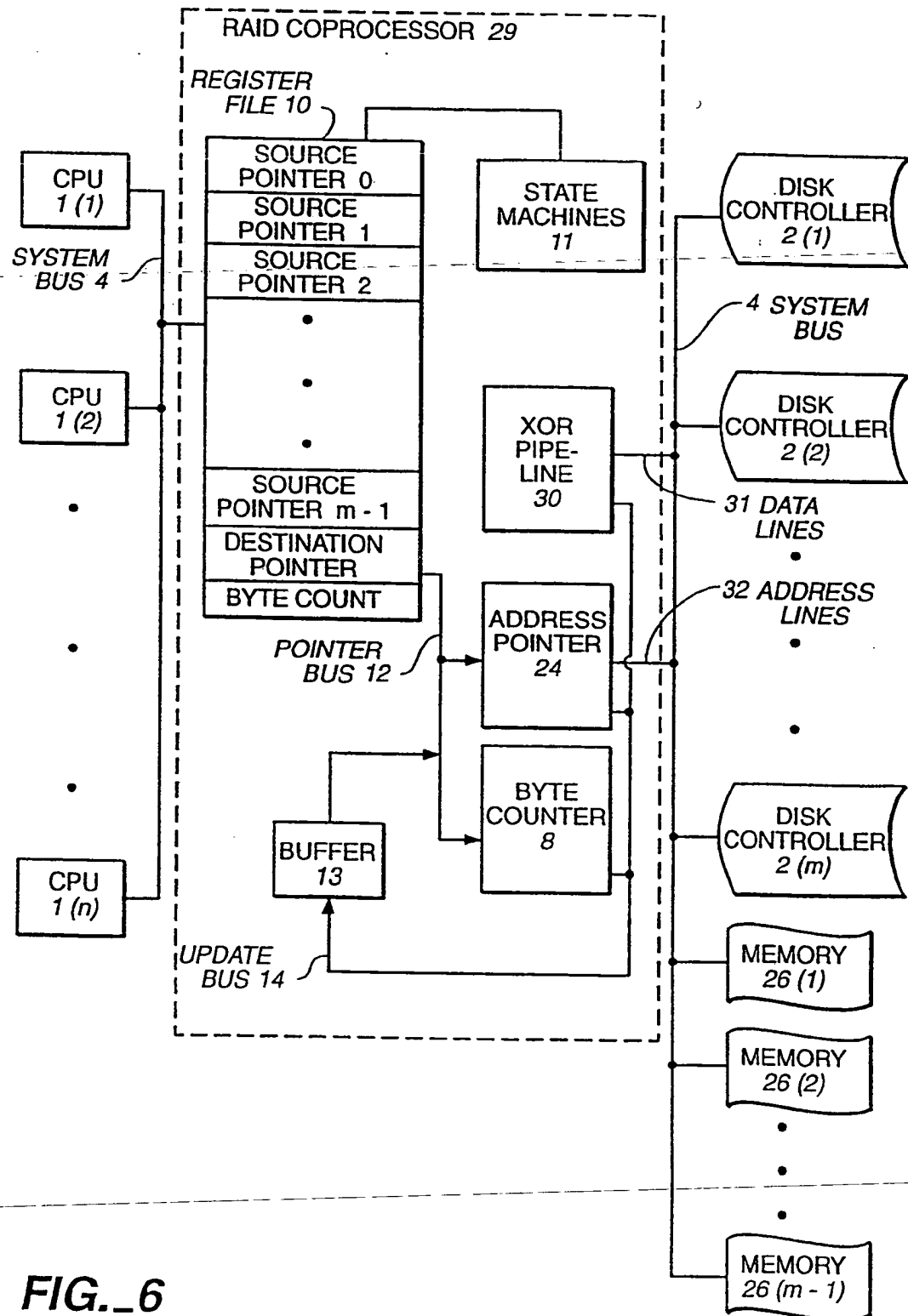


FIG._6

Form PCT/ISA/210 (extra sheet) (January 1985)

**ANNEX TO THE INTERNATIONAL SEARCH REPORT
ON INTERNATIONAL PATENT APPLICATION NO.**

JP 9300617
SA 73160

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information. 06/09/93

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO-A-9111767	08-08-91	US-A- 5175825	29-12-92
		AU-A- 6336990	21-08-91
		EP-A- 0512991	19-11-92
EP-A-0365116	25-04-90	JP-A- 2148245	07-06-90
GB-A-2211325	28-06-89	DE-A- 3835125	03-05-89
		JP-A- 1236342	21-09-89
EP-A-0482819	29-04-92	US-A- 5208813	04-05-93
		AU-A- 8590491	07-05-92
		CA-A- 2053692	24-04-92
		JP-A- 5143471	11-06-93

EPO FORM P0479

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82